Ghostbuster: Detecting Text Ghostwritten by Large Language Models

Vivek Verma **Eve Fleisig** **Nicholas Tomlin** **Dan Klein**
Computer Science Division, University of California, Berkeley
{vivekverma, efleisig, nicholas_tomlin, klein}@berkeley.edu

Abstract

We introduce Ghostbuster, a state-of-the-art system for detecting AI-generated text. Our method works by passing documents through a series of weaker language models and running a structured search over possible combinations of their features, then training a classifier on the selected features to determine if the target document was AI-generated. Crucially, Ghostbuster does not require access to token probabilities from the target model, making it useful for detecting text generated by black-box models or unknown model versions. In conjunction with our model, we release three new datasets of human and AI-generated text as detection benchmarks that cover multiple domains (student essays, creative fiction, and news) and task setups: document-level detection, author identification, and a challenge task of paragraph-level detection. Ghostbuster averages 99.1 F1 across all three datasets on document-level detection, outperforming previous approaches such as GPTZero and DetectGPT by up to 32.7 F1. See https://github.com/vivek3141/ghostbuster for more details.

1 Introduction

Text generation tools such as ChatGPT are capable of producing a wide range of fluent text that closely approximates human-authored text. However, the use of LLMs for classroom assignments raises questions about the authenticity and originality of student work. Concerns that students are submitting assignments *ghostwritten* by language models has led many schools to adapt by restricting the use of ChatGPT and similar models (Heaven, 2023). Several detection frameworks have been proposed to address this issue, such as GPTZero (Tian, 2023) and DetectGPT (Mitchell et al., 2023). While these frameworks offer some level of detection, we find that their performance falters on datasets that they were not originally evaluated on (Section 6). In addition, the high false positive rates of these models raise potential ethical concerns because they jeopardize students whose genuine work is misclassified as AI-generated.

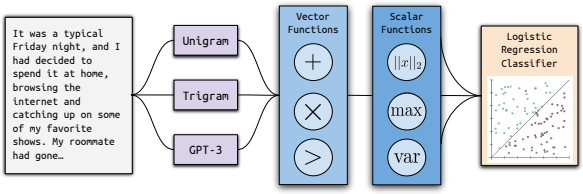

Figure 1: An outline of the classification algorithm. First, we generate possible combinations of features outputted by a sequence of weaker models. Then, we run a structured search over combinations of the model outputs and train a linear classifier on the selected features.

Whether detection of AI-generated text is a trivial or intractable task depends greatly on the framing of the problem. Generated text that contains human paraphrases or was generated from particularly clever prompts is particularly difficult to detect with machine learning methods (Sadasivan et al., 2023). However, trained humans have consistently been able to spot ChatGPT's style (Guo et al., 2023). Thus, we introduce benchmarks for several framings of the detection task, with ascending levels of difficulty: *author identification*, detecting whether a document was AI-generated or written by a single author, given a history of documents by that author; *document-level detection*, detecting whether a full document was AI-generated; and *paragraph-level detection*, detecting which paragraphs in a document were AI-generated.

Our proposed tasks are motivated by real-world applications of these detectors. For example, when questioning whether a student assignment was AI-generated, instructors often have access to previous work by that student; document-level detection may be useful when instructors do not have access to a history of student writing, or cannot verify that students did not include AI-generated text in previ-

ous assignments; and paragraph-level detection is useful when, as is often the case, AI assistance was only used for portions of the assignment.

We also introduce three new datasets for benchmarking detection of AI-generated text across different domains. Our *creative writing* dataset includes human-authored stories from the `r/WritingPrompts` subreddit. Our *news* dataset includes human-authored articles from the Reuters 50-50 dataset (Houvardas and Stamatatos, 2006), which consists of 50 train and 50 test articles for each of 50 authors. Finally, our *student essay* dataset includes student essays from the British Academic Written English Corpus (BAWE) (Nesi et al., 2023). For each document in each dataset, we generate corresponding ChatGPT articles based on the same prompt, summary, or article headline.

In this paper, we present a straightforward method for detection based on structured search and linear classification. First, we pass all documents through a series of weaker language models, ranging from a unigram model to the non-instruction-tuned GPT-3 `davinci`. Given the word probabilities from these models, we search over a space of vector and scalar functions which combine these probabilities into a small set of features. We then feed these features into a linear classifier, as described in Section 4. Averaged across all three datasets, our model achieves 99.1 F1 on document-level identification, outperforming GPTZero and DetectGPT by 9.4 F1. We release code for our method and replicating our experiments at `https://github.com/vivek3141/ghostbuster`.

2 Related Work

AI-generated text exhibits qualitative differences from human-authored text, though these are often subtle. Guo et al. (2023) found that while volunteers often rated ChatGPT answers as more helpful than human ones, ChatGPT answers were more formal, more strictly focused, and used more conjunctions. Jawahar et al. (2020) found that GPT-2 responses that a model misclassified as human-authored tended to be very short and contained issues of factuality, repetition, contradiction, and incoherence. Dou et al. (2022) had human labelers annotate AI-generated text and found that larger models were less likely to produce responses that were factually incorrect, contradicted common sense, or were incoherent, while the degree of self-contradiction and redundant text varies substantially depending on the model.

AI-generated text reflects some, but not all, statistical properties of human-authored text. Meister and Cotterell (2021) find that AI-generated text (from CNN, LSTM, and transformer models) adheres to Heaps's type-token law less than human-authored text, though this improves with nucleus sampling. However, AI-generated text is similar to human text in terms of unigram distribution, length, stopword/symbol usage, and adherence to Zipf's rank-frequency law. Ippolito et al. (2020) argue that improved model decoding strategies, particularly top-k sampling, are better at fooling humans but simultaneously introduce statistical abnormalities, such as different unigram distributions, that make this generated text easier to detect automatically.

Several tools have recently been introduced to detect AI-generated text. Gehrmann et al. (2019) introduced GLTR, a suite of statistical tools to aid humans in detecting AI-generated text, which include overlaying text with the text's top-k annotation in different colors. Uchendu et al. (2020) use a RoBERTA-based model to identify whether two texts are generated by the same method, whether a text is AI-generated, and which of a set of candidate methods generated a text. DetectGPT (Mitchell et al., 2023) uses the fact that unlike human-authored text, generated text lies in regions of the probability space where nearby samples often have lower model probability. It generates random perturbations of the text from a generic LM to detect AI-generated text, then gets probabilities of the original text and perturbation from the model that might have generated the text.

However, Sadasivan et al. (2023) argue that there is an upper bound on the performance of generated text detectors and find that many detectors are brittle to paraphrasing attacks, including DetectGPT (Mitchell et al., 2023), GLTR (Gehrmann et al., 2019), other zero-shot methods (Ippolito et al., 2020; Solaiman et al., 2019), as well as OpenAI's generated text detectors (OpenAI, 2019). In this paper, we focus primarily on the setting in which entire paragraphs or documents were generated by language models, leaving adversarial prompting or paraphrasing-based attacks to future work.

3 Datasets and Detection Tasks

We collected three new datasets for benchmarking detection of AI-generated text across the domains of creative writing, news, and student essays.

Our datasets are constructed to facilitate evaluation on author identification, document-level detection, and paragraph-level detection.

3.1 Datasets

We collected ChatGPT generated text corresponding to the human-authored text for each of the three datasets. All datasets were generated using `gpt-3.5-turbo`, with the exception of the paragraph-level data, which was generated using `text-davinci-003`'s insert feature. All our final datasets are evenly split between human-authored and AI-generated text. For each task, the datasets are divided into train, validation, and test sets.

3.1.1 Creative Writing Dataset

Our creative writing dataset is based on the subreddit `r/WritingPrompts`, a community in which users share creative writing prompts, and also craft stories to address these prompts. In order to avoid contamination from ChatGPT-written content, we collected data from the top 50 posters in October, 2022 and scraped the last 100 posts of each of these users. For each story in the dataset, we generate a corresponding GPT example using the prompt `Write a story in {words} words to the prompt: {prompt}`. We set `words` equal to the number of words in the human-written example rounded to the nearest 50, and `prompt` with the prompt corresponding to each story. This approach is intended to prevent document length or content effects from trivializing the detection task.

3.1.2 News Dataset

Our news dataset is based on the Reuters 50-50 authorship identification dataset (Houvardas and Stamatatos, 2006), which consists of 5000 news articles by 50 journalists. Because we did not have access to ground truth headlines or summaries for these articles, we prompted ChatGPT to generate a headline for each article using the prompt `Create a headline for the following news article: {doc}`. We then prompted ChatGPT to write an article based on each generated headline with `Write a news article in {length} words with the following headline: {headline}`.

3.1.3 Student Essay Dataset

Our student essay dataset is based on the British Academic Written English (BAWE) corpus (Nesi et al., 2023), which consists of 2685 university student essays across a range of disciplines. For each essay in the dataset, we first generate a prompt with `Given the following essay, write a prompt for it: {doc}`, then generate a corresponding example using the prompt `Write an essay in {words} words to the prompt: {prompt}`.

Vector Functions	Scalar Functions
$f_{\text{add}_i} = p_{1_i} + p_{2_i}$	$f_{\text{max}} = \max p$
$f_{\text{sub}_i} = p_{1_i} - p_{2_i}$	$f_{\text{min}} = \min p$
$f_{\text{mul}_i} = p_{1_i} \cdot p_{2_i}$	$f_{\text{avg}} = \frac{1}{\|p\|} \sum_i p_i$
$f_{\text{div}_i} = p_{1_i} / p_{2_i}$	$f_{\text{avg-top-25}} = \frac{1}{\|p\|} \sum_{i \in T_p} p_i$
$f_{>_i} = \mathbb{1}\{p_{1_i} > p_{2_i}\}$	$f_{\text{len}} = \|p\|$
$f_{<_i} = \mathbb{1}\{p_{1_i} < p_{2_i}\}$	$f_{\text{L2}} = \|\|p\|\|_2$
	$f_{\text{var}} = \frac{1}{n} \sum_i (p_i - \mu_p)^2$

Table 1: List of vector and scalar functions used for feature generation. Vector functions take in two vectors of log probabilities $p_1, p_2 \in \mathbb{R}^n$ and output a single vector $f \in \mathbb{R}^n$, where n is the number of tokens in a document. On the other hand, scalar functions take in an input vector $p \in \mathbb{R}^n$ and output $f \in \mathbb{R}$. Here, T_p denotes the indices that contain the top 25 lowest values in p and μ_p denotes the average value of p.

3.2 Detection Tasks

For document-level identification, we evaluate the F1 score, accuracy, and AUROC (area under the receiver operating characteristic) of our model at classifying the text in each dataset as human-authored or AI-generated. Author identification performance was evaluated on the news and creative writing datasets, as the student essay dataset does not link essays to their authors. We train separate classifiers for each author and evaluate the F1 score, accuracy, and AUROC of determining whether each document was written by that author or AI-generated.

4 Methods

During training, Ghostbuster first passes each document through a series of weaker language models to compute token log probabilities for each document. Our approach uses a unigram model, a KN trigram model, and two early GPT-3 models (`ada` and `davinci`, without instruction tuning) to get these probabilities. Given these probabilities, we run a structured search procedure over a space of vector and scalar functions that combine these probabilities. To do so, we define a set of operations that combine these features, run forward feature selection on them, and train a simple classifier on the

best probability-based features and some additional manually-selected features.

4.1 Feature Selection

Feature selection proceeds in two stages: we first generate a set of features (Table 1), then combine them using Algorithm 1. To generate features, we first outline the 13 scalar and vector functions in Table 1, where scalar functions convert vector to scalars, and vector functions combine two vectors into one. In order to generate all possible features, we run Algorithm 1 four times, with the log probability vectors from each model as the starting features and a maximum depth of 3. Features thus take the form of combining three arbitrary log probabilities with vector functions, then reducing them to a scalar function. An example feature is `var(unigram-logprobs > ada-logprobs - davinci-logprobs)`. We provide more details on the algorithm and its implementation and outputs in Appendix A. For a version of Ghostbuster trained on each dataset, we run forward feature selection to find the best features, which are listed in Appendix C.

Algorithm 1 Subroutine FIND-ALL-FEATURES

Require: The previous feature p being worked with, depth $d \leq$ `max_depth`, set V of vectors (unigram, trigram, `ada`, `davinici`)
Ensure: A list of all possible features
  Let S be an empty set of features
  for all scalar functions f_s **do**
    Add $f_s(p)$ to S
  end for
  for all combinations of $p' \in V$ and functions f_v **do**
    Add FIND-ALL-FEATS$(f_v(p, p'), d + 1)$ to S
  end for

4.2 Classifier

Our approach consists of a linear classifier trained on combinations of the probability-based features chosen through structured search, and seven additional features (Appendix B) based on word length and the largest log probabilities. These additional features are intended to incorporate qualitative heuristics observed about AI-generated text: that text is "uninformative," exhibiting patterns of surprisal that differ from human-authored text in ways that may be evident in the frequency of outliers or differences in log probabilities between tokens that the models rate as very likely or only moderately likely; and that AI-generated text may have a tendency to generate words that are split into fewer subword tokens than human-authored text.

We then train a logistic regression classifier with l_2 regularization and setting $C = 10$ that takes in these features and those chosen through structured search (Section 4.1).

5 Baseline Models

We evaluated GPTZero (Tian, 2023), DetectGPT (Mitchell et al., 2023) and zero-shot ChatGPT on the same tasks as our models for comparison. In order to improve the performance of baseline models and provide a fair comparison, we calibrate the output probabilities of GPTZero and DetectGPT. We note that the results provided by our model are *not* calibrated with this method. When running DetectGPT, we evaluated all data with base models GPT-2 XL, Ada, and Davinci, and chose the best result out of these three. For the zero-shot case, we prompt ChatGPT with `Was this text written by ChatGPT?` followed by the text.

We note that the baseline model results are zero-shot, since the models have not seen any of the training data, which may lead to an unfair comparison with Ghostbuster. To address this, in addition to calibrating the baseline models, we also provide out-of-domain generalization results, where we train Ghostbuster on one dataset (news, essays, or stories) and evaluate on the others. One limitation of this approach is that all our datasets were generated from ChatGPT and use similar prompting methods; thus, a remaining d of this comparison is that it is unclear whether the degree of domain shift between Ghostbuster's training data and this test data is similar to the degree of domain shift between the baseline models' training data and the test data. In our results (Table 2), we provide both the best out-of-domain generalization result for testing on each dataset (training on a different dataset), and the in-domain result for testing on each dataset (training on the same dataset).

6 Results

We provide results across all three datasets in Table 2. We also provide accuracy and AUROC values in Appendix D. As the zero-shot ChatGPT baseline performs worse than all other methods tested for document-level detection and author identifica-

Method	News			Creative Writing			Student Essay	
	Document	Author	Paragraph	Document	Author	Paragraph	Document	Para
GPTZero	93.92	93.92	49.65	90.64	90.64	**61.81**	84.10	36.70
DetectGPT	64.48	64.48	37.74	68.01	68.01	59.44	77.50	45.63
Zero-Shot	44.14	44.14	**54.74**	28.78	28.78	20.00	53.43	**52.29**
Ghostbuster (out-of-domain)	98.28	–	34.01	98.21	–	49.53	98.21	51.21
Ghostbuster (in-domain)	**99.01**	**99.13**	52.01	**99.26**	**98.70**	41.13	**99.07**	42.55

Table 2: Results of evaluating methods on each of our datasets (F1). Out-of-domain results are computed by taking the maximum performance of our model when trained on one of the two other datasets. In-domain results are computed by training only on the target dataset. We do not compute out-of-domain author identification results.

tion, and performs comparably to all models tested for paragraph-level detection, we center our analysis on the comparisons to DetectGPT.

On document-level detection, our model scores 99.1 F1 averaged across all datasets when evaluated in-domain, outperforming DetectGPT by an average margin of 28.9, and GPTZero by 9.4. When evaluated out-of-domain (trained on one dataset and evaluated on another), Ghostbuster scores 98.2 F1 averaged across all datasets, outperforming DetectGPT by an average margin of 28.2 and GPTZero by 8.6. These results suggest that Ghostbuster's performance gains are robust with respect to the similarity of the training and testing datasets. On author identification, our model scores 98.9 F1 (averaged across news and creative writing) when evaluated in-domain, outperforming DetectGPT by an average margin of 32.7 and GPTZero by 6.6.

Paragraph-level detection constitutes a challenge task for which we expect lower model performance. Our model scores 45.2 F1 in-domain (averaged across all datasets) and 44.91 F1 out-of-domain, compared to DetectGPT scoring 47.6 F1, GPTZero scoring 49.4 F1, and zero-shot ChatGPT scoring 42.3 F1. For this task, no model is consistently the best across all domains. These low scores across the board suggest that paragraph-level detection is a substantially more difficult task that is a key area for future work.

In addition, we evaluated our classifier when selecting 7 random features for 100 iterations, achieving a score of 85.6 F1. This provides evidence for the neccessity of the structured search along with feature selection.

7 Discussion

We find that Ghostbuster predicts whether a document was AI-generated, given the document alone or compared to a history of previous documents by an author. Both in-domain or out-of-domain, Ghostbuster has upwards of 98.1 F1 across all datasets and and improves over previous work by 5.5 to 32.7 F1. Our results suggest that combining features of log probabilities of weaker models helps to accurately identify text generated by stronger models without access to probabilities from those models. This finding opens avenues for using Ghostbuster in situations with constraints that are common in real-world situations: lack of access to probabilities from a black-box model that may have generated a piece of text, uncertainty about which model was used to generate a text, or resource constraints that make retrieving model probabilities from the largest available models too expensive.

The generally low results for both our model and the baselines tested suggest that paragraph-level detection of AI-generated text is a challenging task. This is unsurprising, as it requires accurate detection on lower numbers of tokens and for multiple paragraphs within a document. However, it does mean that accurate paragraph-level detection is a key problem for future work. The datasets that we propose introduce benchmarks for benchmarking progress on this task, as well as the comparatively less difficult tasks of document-level detection and author identification, across the domains of news, creative writing, and student essays.

8 Conclusion

We introduced Ghostbuster, a model for detecting AI-generated text that uses structured search on token probabilities from weaker models to identify whether text was AI-generated. We validated Ghostbuster by evaluating its performance on datasets from three domains (news, student essays, and creative fiction writing) and with three task

framings (document-level detection, author identification, and the challenge paragraph-level detection task). We also provide our three datasets across different domains as benchmarks for evaluating performance on detecting AI-generated text. Ghostbuster achieves over 98.1 F1 across all datasets on document-level detection and author identification, representing substantial progress over currently available models for detection of AI-generated text. Meanwhile, paragraph-level detection remains a major unsolved task for which the datasets introduced here may serve as benchmarks.

Because Ghostbuster does not require access to target model probabilities, it is well-suited to identifying text generated from black-box or unknown models. In addition, the effectiveness of structured search over features of model probabilities at identifying model-generated text suggests that further work extending these techniques to incorporate different potential features could help to make further progress at identification of AI-generated text.

9 Limitations and Ethical Considerations

Our datasets were all generated from ChatGPT using similar prompting methods, so a limitation of our comparison with previous work is that it is unclear whether the degree of domain shift between Ghostbuster's training data and our test data is similar to the degree of domain shift between the baseline models' training data and our test data.

We evaluate Ghostbuster on three datasets that represent a range of domains, but note that these datasets are not representative of all writing styles or topics and contain predominantly British and American English text. Thus, users wishing to apply Ghostbuster to real-world cases of potential off-limits usage of text generation (e.g., identifying ChatGPT-written student essays) should be wary that no model is infallible, and incorrect predictions by Ghostbuster are particularly likely when the text involved is shorter or in domains that are further from those examined in this paper. To avoid perpetuation of algorithmic harms due to these limitations, we discourage incorporation of Ghostbuster into any systems that automatically penalize students or other writers for alleged usage of text generation without human supervision.

10 Acknowledgments

This work was supported by grants from Open Philanthropy and DARPA SemaFor. Many thanks to Lucy Li for her help with the r/writingprompts data collection for this paper; we are also grateful to Ruiqi Zhong, Dan Klein, Mitchell Stern, and Matt Gardner for their feedback and recommendations.

References

Dou, Y., Forbes, M., Koncel-Kedziorski, R., Smith, N. A., and Choi, Y. (2022). Is GPT-3 text indistinguishable from human text? scarecrow: A framework for scrutinizing machine text.

Gehrmann, S., Strobelt, H., and Rush, A. M. (2019). GLTR: Statistical detection and visualization of generated text.

Guo, B., Zhang, X., Wang, Z., Jiang, M., Nie, J., Ding, Y., Yue, J., and Wu, Y. (2023). How close is ChatGPT to human experts? comparison corpus, evaluation, and detection.

Heaven, W. (2023). Chatgpt is going to change education, not destroy it. *MIT Technology Review*.

Houvardas, J. and Stamatatos, E. (2006). N-gram feature selection for authorship identification. In *Artificial Intelligence: Methodology, Systems, Applications*.

Ippolito, D., Duckworth, D., Callison-Burch, C., and Eck, D. (2020). Automatic detection of generated text is easiest when humans are fooled.

Jawahar, G., Abdul-Mageed, M., and Lakshmanan, L. V. S. (2020). Automatic detection of machine generated text: A critical survey.

Meister, C. and Cotterell, R. (2021). Language model evaluation beyond perplexity.

Mitchell, E., Lee, Y., Khazatsky, A., Manning, C. D., and Finn, C. (2023). DetectGPT: Zero-shot machine-generated text detection using probability curvature.

Nesi, H., Gardener, S., Alsop, S., Thompson, P., Wickens, P., Leedham, M., and Ebeling, S. O. (2023). (bawe) british academic written english corpus.

OpenAI (2019). GPT-2: 1.5b release.

Sadasivan, V. S., Kumar, A., Balasubramanian, S., Wang, W., and Feizi, S. (2023). Can ai-generated text be reliably detected?

Solaiman, I., Brundage, M., Clark, J., Askell, A., Herbert-Voss, A., Wu, J., Radford, A., Krueger, G., Kim, J. W., Kreps, S., McCain, M., Newhouse, A., Blazakis, J., McGuffie, K., and Wang, J. (2019). Release strategies and the social impacts of language models.

Tian, E. (2023). GPTZero: Home.

Uchendu, A., Le, T., Shu, K., and Lee, D. (2020). Authorship attribution for neural text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8384–8395, Online. Association for Computational Linguistics.

A Algorithm Implementation Details

In this section, we describe more details of the algorithm that generates the feature space. While the algorithm presented in Algorithm 1 produces equivalent results to our implementation, we make a few additional optimizations. First, we note that all vector functions in Table 1 are commutative, or posses the inverse operator. As such, we first create a list of possible vector combinations, avoiding double-counting. We noticed that this pruning results in around a 2/3rd reduction in the feature space. At a depth of 3, we have 2534 features, and at a depth of 2 we have 322 features.

B Additional Features

Ghostbuster uses the following handcrafted features in addition to those chosen through feature selection:

- Number of outliers ($p_i > 10$), average value of top 25 and 25-50$^{\text{th}}$ largest log probabilities
- Average value of the 25 largest and 25-50$^{\text{th}}$ largest log probabilities of the vector $d - a$, where d is a vector of Davinci log probabilities and a is a vector of Ada log probabilities.
- Average length of the 25 longest and 25-50$^{\text{th}}$ longest words, measured in tokens.

C Best Features for Each Dataset

In this section, we present the best features chosen through validation on each of the datasets. For a list of functions and features used, refer to Table 1.

C.1 Creative Writing Dataset

```
avg(unigram + ada + davinci)
var(unigram > ada - davinci)
avg(unigram)
min(ada)
avg(davinci / trigram * ada)
min(unigram > davinci)
```

C.2 News Dataset

```
var(unigram > ada)
avg(ada - davinci)
avg(unigram * ada)
avg(trigram < ada)
var(unigram < trigram)
```

C.3 Student Essay Dataset

```
var(unigram > ada > davinci)
avg(unigram + trigram > davinci)
avg(unigram + ada - davinci)
min(davinci / trigram + davinci)
```

D More Results

We provide AUROC and Accuracy values in Table 3 and Table 4. Similar to the F1 scores, they show Ghostbuster improves previous work by a factor of 7-19%. In addition, we also include ROC curves for in-domain predictions in Figure 2.

	News			Creative Writing			Student Essay	
Method	Document	Author	Paragraph	Document	Author	Paragraph	Document	Para
GPTZero	0.9748	0.9748	0.3904	0.9664	0.9664	0.5229	0.9335	0.4594
DetectGPT	0.6837	0.6837	0.4260	0.7556	0.7556	0.5801	0.8392	0.5452
Ghostbuster (out-of-domain)	0.9971	–	0.5389	0.9963	–	0.6981	0.9968	0.6466
Ghostbuster (in-domain)	**0.9994**	**0.9991**	**0.6631**	**0.9993**	**0.9969**	**0.6907**	**0.9991**	**0.6485**

Table 3: AUROC when evaluating various methods on our proposed datasets. "Out of domain" for a dataset D denotes a model evaluated on the test split of D, but trained on the train split of a different dataset. "In domain" denotes a model trained on D's train split and evaluated on D's test split.

	News			Creative Writing			Student Essay	
Method	Document	Author	Paragraph	Document	Author	Paragraph	Document	Para
GPTZero	93.85	93.85	41.60	90.70	90.70	53.80	84.09	50.62
DetectGPT	64.21	64.21	44.75	68.00	68.00	56.00	77.50	51.75
Zero-Shot	43.22	43.22	43.22	49.61	49.61	49,61	46.42	46.42
Ghostbuster (out-of-domain)	98.20	–	55.89	98.20	–	**66.91**	98.21	**63.62**
Ghostbuster (in-domain)	**99.02**	**99.14**	64.89	**99.25**	**98.72**	63.61	**99.07**	**61.72**

Table 4: Accuracy of evaluating various methods on our proposed datasets. "Out of domain" for a dataset D denotes a model evaluated on the test split of D, but trained on the train split of a different dataset. "In domain" denotes a model trained on D's train split and evaluated on D's test split.

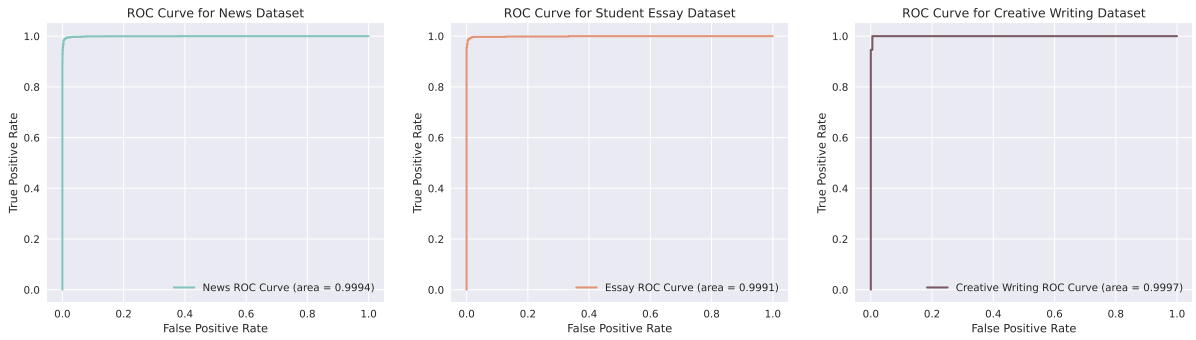

Figure 2: ROC curves for each of the three datasets, evaluated in-domain.